CLAIMS

What is claimed is:

1. A processor that includes an in-order execution architecture for executing at least two instructions per cycle and at least two symmetric execution units comprising:

a) instruction fetch unit for fetching n instructions;

b) an instruction decoder for decoding the n instruction;

wherein 2n instructions are processed per cycle

c) duplication hardware for duplicating the n instructions into a first bundle and a second bundle; wherein each bundle includes n instructions;

d) a first execution unit for executing the first bundle of instructions in a first execution cycle;

e) the second symmetric execution unit for executing the second bundle of instructions in the first execution cycle;

f) comparison hardware for comparing the results of the first execution unit and the results of the second execution unit; and

g) a commit unit for committing one of the results when the results are the same; and

h) an exception unit for generating an exception (raising a fault) when the results are not the same.

2. The processor of claim 1

wherein the first execution unit issues the first bundle of instructions to the first execution unit; and

wherein the second symmetric execution unit issues the second bundle of instructions to the second execution unit in the first execution cycle.

3. The processor of claim 2

wherein the first execution unit is one of floating point unit, an integer unit, a arithmetic logic unit (ALU), a multimedia unit, and a branch unit; and

wherein the second execution unit is symmetric with respect to the first execution unit and includes one of floating point unit, an integer unit, a arithmetic logic unit (ALU), a multimedia unit, and a branch unit.

4. The processor of claim 1 wherein duplication hardware is provided for performing the instruction duplication and comparison hardware is provided for performing the comparison, the method further comprising the step of:

setting a bit in a control register;

wherein the bit enables the duplication hardware and comparison hardware.

5. The processor of claim 4 wherein the bit is set by one of user-programmed firmware, an operating system (OS), and an application.

6. The processor of claim 1 wherein n is equal to 3.

7. A method for detecting errors in a processor that executes 2n instructions per cycle comprising the steps of:

a) fetching n instructions; wherein n is an integer greater than 0;

b) decoding the n instructions;

c) duplicating the n decoded instructions into a first bundle of n decoded instructions and a second bundle of n decoded instructions;

d) employing a first execution unit to execute the first bundle of instructions in a first execution cycle;

e) employing a second symmetric execution unit for executing the second bundle of instructions in the first execution cycle;

f) comparing the results of the first execution unit and the results of the second execution unit;

g) when the results are the same, committing one of the results; and

h) when the results are not the same, generating an exception (raising a fault).

8. The method of claim 7

wherein the step of employing a first execution unit to execute the first bundle of instructions in a first execution cycle includes issuing the first bundle of instructions to the first execution unit; and

wherein the step of employing a second symmetric execution unit for executing the second bundle of instructions in the first execution cycle includes issuing the second bundle of instructions to the second execution unit.

9. The method of claim 7

wherein the first execution unit is one of floating point unit, an integer unit, a arithmetic logic unit (ALU), a multimedia unit, and a branch unit; and

wherein the second execution unit is symmetric with respect to the first execution unit and one of floating point unit, an integer unit, a arithmetic logic unit (ALU), a multimedia unit, and a branch unit.

10. The method of claim 7 wherein duplication hardware is provided for performing the instruction duplication and comparison hardware is provided for performing the comparison, the method further comprising the step of:

setting a bit in a control register;

wherein the bit enables the duplication hardware and comparison hardware.

11. The method of claim 10 wherein the bit is set by one of user-programmed firmware, an operating system (OS), and an application.

12. The method of claim 7 wherein each instruction includes a bit for enabling the instruction for error checking.

13. The method of claim 7 wherein n is equal to 3.

14. A method for selectively enabling an error detection mechanism comprising the steps of:

a) maintaining a control register that includes an error detection enable bit;

b) setting the error detection enable bit to enable the error detection mechanism; and

c) clearing the error detection enable bit to disable the error detection mechanism.

15. The method of claim 14 wherein the step of setting the error detection enable bit to enable the error detection mechanism includes one of

a user-programmed firmware setting the error detection enable bit to enable the error detection mechanism;

an operating system setting the error detection enable bit to enable the error detection mechanism; and

an application setting the error detection enable bit to enable the error detection mechanism; and

wherein the step of clearing the error detection enable bit to disable the error detection mechanism includes one of

a user-programmed firmware clearing the error detection enable bit to enable the error detection mechanism;

an operating system setting clearing the error detection enable bit to enable the error detection mechanism; and

an application clearing the error detection enable bit to enable the error detection mechanism.

16. The method of claim 14 wherein the error detection mechanism is enabled for a portion of critical code that includes a first instruction and a last instruction;

wherein the step of setting the error detection enable bit to enable the error detection mechanism includes the step of

setting the error detection enable bit to enable the error detection mechanism prior to the execution of the first instruction of the critical portion of code; and

wherein clearing the error detection enable bit to disable the error detection mechanism includes

clearing the error detection enable bit to disable the error detection mechanism after the execution of the last instruction of the critical portion of code.

17. An apparatus for executing instructions comprising:

a) a control register that includes an error detection enable bit;

b) an error detection mechanism for detecting soft errors; and

c) a mechanism for selectively enabling the error detection mechanism by setting the error detection enable bit to enable the error detection mechanism and by clearing the error detection enable bit to disable the error detection mechanism.

18. The apparatus of claim 17 wherein the selective enabling mechanism is one of a user-programmed firmware, an operating system, and an application.

19. The apparatus of claim 17 wherein the error detection mechanism is enabled for a portion of critical code that includes a first instruction and a last instruction;

wherein the selective enabling mechanism sets the error detection enable bit to enable the error detection mechanism prior to the execution of the first instruction of the critical portion of code; and

wherein the selective enabling mechanism clears the error detection enable bit to disable the error detection mechanism after the execution of the last instruction of the critical portion of code.